CompSci 101 AC 2002 Terms Test

MODEL ANSWERS

QUESTION 1

a) `2ter`
b) `3.34`
c) THE CODE WILL COMPILE
d)
```
12
3
4
56
```
e) `System.out.println("\\n\\\n\\\\n");`
f) `i is: 80`
g) `d is: 4.5`
h) `-165`
i) `5`

QUESTION 2

4 syntax errors:

1) `String allStars = '**********************************';`
   single quotes, should be double quotes
2) `System.out.println(stars + " " + s + " " + stars);`
   missing concatenation operator after the `s`
3) `public static void main(String[] args) {`
   missing word `void`
4) `printWithStars(name, number);`
   variable `number` should be `num`

1 logic error:

5) `String stars = allStars.substring(0, numStars-1);`
   `numStars-1` should be `numStars`

QUESTION 3

```
private static int middleNumber(int a, int b, int c) {
     int max = Math.max(a, Math.max(b, c));
     int min = Math.min(a, Math.min(b, c));
     return (a+b+c) - (max+min);
}
```

QUESTION 4

```
2.0
no: 10.1
no: 21.5
100.0
```

QUESTION 5

```
private static int randomOddDigit() {
     int num = (int)(Math.random() * 5);
     return num*2 + 1;
}
```

QUESTION 6

```
Line 5
Line 6
Line 7
Line 9
Line 10
```

QUESTION 7

```
public static void printLuckyColour(){
  int rand = (int) (Math.random()*7);
  if (rand==0 || rand==1)
     System.out.println("Lucky colour: green");
   else if (rand==2 || rand==3 || rand==4)
     System.out.println("Lucky colour: red");
   else
     System.out.println("Lucky colour: blue");
}
```

QUESTION 8

```
b1: true
b2: true
b3: false
```

QUESTION 9

```
public static void printColumn(int num){
   int downCounter = num;
   int upCounter = 1;
   while (downCounter>0){
     System.out.println(downCounter + " - " + upCounter);
     upCounter++;
     downCounter--;
   }
}
```

QUESTION 10

```
gentle
breathing
```

QUESTION 11

```
 public class Locker{

 private int num, floorNum;
 private boolean isVacant;
 private String hirer;
//-------------------------------------------------------------
// Constructor
//-------------------------------------------------------------
 public Locker(String hire, int floor, int num){
   this.num = num;
   floorNum = floor;
   hirer = hire;
   isVacant = false;
 }
//-------------------------------------------------------------
// Return isVacant
//-------------------------------------------------------------
 public boolean getIsVacant(){
    return isVacant;
 }
//-------------------------------------------------------------
// set the locker to vacant
//-------------------------------------------------------------
 public void setIsVacant(boolean b){
    isVacant = b;
    if (isVacant)
       hirer = "";  //optional
 }
//-------------------------------------------------------------
// Return true if the two lockers have the same hirer
//-------------------------------------------------------------
 public boolean hasSameHirer(Locker otherL){
   if (hirer.equals(otherL.hirer))
      return true;
   return false;
 }
//-------------------------------------------------------------
// Return description of the locker
//-------------------------------------------------------------
 public String toString(){
   String str = "Floor "+floorNum+", Locker
number: "+num;
   if (isVacant)
      str += "  Currently vacant.";
   else
      str += "  Currently hired by: "+hirer;
   return str;
 }
}
```